

RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	000	FFFFFFFFFFF	FFFFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	000	FFFFFFFFFFF	FFFFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	000	FFFFFFFFFFF	FFFFFFFFFFF
RRR	RRR	UUU	UUU	NNN	NNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	000	FFF	FFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	000	FFFFFFFFF	FFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	000	FFFFFFFFF	FFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	000	FFFFFFFFF	FFFFFFFFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	000	FFF	FFF
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	NNN	000000000	000	FFF	FFF
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	NNN	000000000	000	FFF	FFF
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	NNN	000000000	000	FFF	FFF

```
EEEEEEEEEE RRRRRRRR RRRRRRRR 000000 RRRRRRRR
EEEEEEEEEE RRRRRRRR RRRRRRRR 000000 RRRRRRRR
EE          RR      RR RR      RR 00      00 RR      RR
EE          RR      RR RR      RR 00      00 RR      RR
EE          RR      RR RR      RR 00      00 RR      RR
EE          RR      RR RR      RR 00      00 RR      RR
EEEEEEEEEE RRRRRRRR RRRRRRRR 00      00 RRRRRRRR
EEEEEEEEEE RRRRRRRR RRRRRRRR 00      00 RRRRRRRR
EE          RR      RR RR      RR 00      00 RR      RR
EE          RR      RR RR      RR 00      00 RR      RR
EE          RR      RR RR      RR 00      00 RR      RR
EEEEEEEEEE RR      RR RR      RR 00      00 RR      RR
EEEEEEEEEE RR      RR RR      RR 000000 RR      RR
                    000000
                    RR      RR
                    RR      RR
                    RR      RR
                    RR      RR
                    ....
                    ....
                    ....
                    ....
```

```
LL          IIIIII SSSSSSSS
LL          IIIIII SSSSSSSS
LL          II      SS
LL          II      SS
LL          II      SS
LL          II      SS
LL          II      SSSSSS
LL          II      SSSSSS
LL          II      SS
LL          II      SS
LL          II      SS
LL          II      SS
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS
```

```
0001 0 %TITLE 'Field (error) message requests from other modules'
0002 0 MODULE error ( IDENT = 'V04-000'
P 0003 0 %BLISS32[, ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE,
0004 0 NONEXTERNAL = LONG_RELATIVE)]
0005 0 ) =
0006 1 BEGIN
0007 1
0008 1 *****
0009 1 *
0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 * ALL RIGHTS RESERVED.
0013 1 *
0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 * TRANSFERRED.
0020 1 *
0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 * CORPORATION.
0024 1 *
0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *
0029 1 *****
0030 1
0031 1 ++
0032 1 FACILITY: DSR (Digital Standard RUNOFF) / DSRPLUS
0033 1
0034 1 ABSTRACT:
0035 1
0036 1 This module generates messages (generally errors) for RUNOFF.
0037 1 These messages are edited at the time the routine is called to
0038 1 allow inclusion of various information necessary in the
0039 1 diagnostic. The details of this editing are described below.
0040 1
0041 1 ENVIRONMENT: Transportable
0042 1
0043 1 AUTHOR: D. Knight CREATION DATE: June 1978
0044 1
0045 1
```



ERROR  
V04-000

Field (error) message requests from other modul  
Revision History

E 2  
16-Sep-1984 00:25:59  
14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1 Page 2 (2)

```

: 47 0046 1 %SBTTL 'Revision History'
: 48 0047 1
: 49 0048 1 MODIFIED BY:
: 50 0049 1
: 51 0050 1 014 KAD00014 Keith Dawson 3-May-1983
: 52 0051 1 Support suppression on counting new error messages: RNFCPC2,
: 53 0052 1 PC3, PC4, PC5, CRU, CR1, CR2).
: 54 0053 1
: 55 0054 1 013 KAD00013 Keith Dawson 11-Apr-1983
: 56 0055 1 Support new termination messages (RNFCPC2, PC3, PC4, PC5) for
: 57 0056 1 information written to the .BRN file (and for DSRPLUS cross-
: 58 0057 1 referencing, also read from the input .BRN file). The support
: 59 0058 1 here is the addition of two new %chars:
: 60 0059 1
: 61 0060 1 %A ==> spec of old (input) .BRN file
: 62 0061 1 %B ==> spec of new (output) .BRN file
: 63 0062 1
: 64 0063 1 012 RER00012 Ron Randall 07-Mar-1983
: 65 0064 1 Global edit of all modules. Updated module names, idents,
: 66 0065 1 copyright dates. Changed require files to BLISS library.
: 67 0066 1
: 68 0067 1 --
: 69 0068 1
```

```
.. 71 0069 1 %SBTTL 'Module Level Declarations'
.. 72 0070 1
.. 73 0071 1
.. 74 0072 1 : TABLE OF CONTENTS:
.. 75 0073 1
.. 76 0074 1 FORWARD ROUTINE
.. 77 0075 1   erm: NOVALUE,      ! Error message writer
.. 78 0076 1   erma: NOVALUE,    ! Report error, text of command, & where found
.. 79 0077 1   ermb: NOVALUE,    ! Report error, input text line, & where found
.. 80 0078 1   erme: NOVALUE,    ! Report error extended (w/secondary message)
.. 81 0079 1   erml: NOVALUE,    ! Report error & where in the text it was found
.. 82 0080 1   ermn: NOVALUE,    ! Report error with numeric param & where found
.. 83 0081 1   erms: NOVALUE,    ! Report error with string param & where found
.. 84 0082 1   putmsg: NOVALUE,  ! Miscellaneous messages writer
.. 85 0083 1   scnmsg,          ! Do text insertion.
.. 86 0084 1   report_secondary_message; ! Fetch and report secondary error message
.. 87 0085 1
.. 88 0086 1
.. 89 0087 1 : INCLUDE FILES:
.. 90 0088 1
.. 91 0089 1 LIBRARY 'NXPOR:XPORT';      ! XPORT Library
.. 92 0090 1 REQUIRE 'REQ:RNODEF';      ! RUNOFF variant definitions
.. 93 0221 1
.. 94 U 0222 1 %IF DSRPLUS %THEN
.. 95 U 0223 1 LIBRARY 'REQ:DPL.LIB';    ! DSRPLUS BLISS Library
.. 96 0224 1 %ELSE
.. 97 0225 1 LIBRARY 'REQ:DSRLIB';      ! DSR BLISS Library
.. 98 0226 1 %FI
.. 99 0227 1
100 0228 1
101 0229 1 : OWN STORAGE:
102 0230 1
103 0231 1 LITERAL work_buffer_size = 500;
104 0232 1
105 0233 1 OWN
106 0234 1   work_buffer :                ! Module wide character buffer
107 0235 1   VECTOR [CH$ALLOCATION (work_buffer_size)],
108 P 0236 1   work_desc : $STR_DESCRIPTOR ( ! & associated string desc.
109 P 0237 1   CLASS=FIXED
110 P 0238 1   ,STRING = (work_buffer_size
111 0239 1   ,CH$PTR (work_buffer) ) );
112 0240 1 OWN
113 0241 1   w_buffer1 :                ! Module wide character buffer
114 0242 1   VECTOR [CH$ALLOCATION (130)],
115 P 0243 1   w_desc1 : $STR_DESCRIPTOR ( ! & associated string desc.
116 P 0244 1   CLASS=FIXED
117 P 0245 1   ,STRING = (130
118 0246 1   ,CH$PTR(w_buffer1)));
119 0247 1
120 0248 1
121 0249 1 : EXTERNAL REFERENCES:
122 0250 1
123 0251 1 EXTERNAL
124 0252 1   gca: gca_definition,
125 0253 1   ira: fixed_string,
126 0254 1   irac: irac_definition,
127 0255 1   msgtxt: VECTOR,          ! Vector of CH$PTRs to RUNOFF's messages.
```

ERROR  
V04-000

Field (error) message requests from other modul  
Module Level Declarations

G 2  
16-Sep-1984 00:25:59  
14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1

Page 4  
(3)

```

128      0256 1      pagen: page_definition,
129      0257 1      phan:  phan_definition,
130      0258 1      rnoiob: REF $XPO_IOB (! Output file (document being built)
131      U 0259 1      %IF DSRPLUS %THEN
132      U 0260 1      brnoiob: $XPO_IOB (! Input .BRN file
133      0261 1      %FI
134      0262 1      brnoiob: $XPO_IOB (! Output .BRN file
135      0263 1      semcod,      ! Secondary error message code
136      0264 1      ttoiob: $XPO_IOB (! Standard messages (TTY: usually)
137      0265 1      tteiob: $XPO_IOB (! Standard error unit (TTY: usually)
138      0266 1
139      0267 1      EXTERNAL LITERAL      !Error messages
140      0268 1      rnfloc; ! I - on output page %P; on input line %C of page %I of file "%F"
141      0269 1
142      0270 1      EXTERNAL ROUTINE
143      0271 1      endcmt,      ! Skip to end of comment
144      0272 1      pacbas,      ! Convert binary to specified base
145      0273 1      pacpag,      ! Convert page number to ASCII
146      0274 1      pacstr,      ! Move ASCII characters.
147      0275 1
148      0276 1      %IF %BLISS(BLISS32) %THEN
149      0277 1      sys$getmsg,      ! System routine to fetch message text
150      0278 1      emsg;      ! Error message handler for VMS
151      U 0279 1      %ELSE
152      U 0280 1      xpo$xmsg;      ! XPORT's routine to fetch its msg. text
153      0281 1      %FI
154      0282 1
```

ERR  
V04

: R

:



ERROR  
V04-000

Field (error) message requests from other modul 16-Sep-1984 00:25:59  
ERMA -- add DIRECTIVE and location to error mes 14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742 Page 5  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1 (4)

```
156 0283 1 %SBTTL 'ERMA -- add DIRECTIVE and location to error message'
157 0284 1 GLOBAL ROUTINE erma (error,position) : NOVALUE =
158 0285 1
159 0286 1 ++
160 0287 1 FUNCTIONAL DESCRIPTION:
161 0288 1
162 0289 1 This routine contains a sequence of code that is commonly used
163 0290 1 for handling errors.
164 0291 1
165 0292 1 If the POSITION parameter is TRUE, it calls ENDCMT to
166 0293 1 position to the end of the erroneous command. Then it passes
167 0294 1 to ERM the specified error message (ERROR), and the start and
168 0295 1 length of the string to be inserted into the error message.
169 0296 1 Note that this call on ERM does not depend on the value of POSITION.
170 0297 1
171 0298 1 Finally, it calls ERM to output the error message indicating
172 0299 1 where in the input and output files the error occurred.
173 0300 1
174 0301 1 It is by no means necessary to call ERM only via ERMA. ERMA is
175 0302 1 simply a collection of commonly used code and is here for the
176 0303 1 sake of convenience.
177 0304 1
178 0305 1 FORMAL PARAMETERS:
179 0306 1
180 0307 1 ERROR - Address of desired error diagnostic. Passed to ERM.
181 0308 1 POSITION - If TRUE, ERMA positions to the end of the
182 0309 1 command before issuing the second call on ERM.
183 0310 1
184 0311 1 IMPLICIT INPUTS:
185 0312 1
186 0313 1 RNFLOC - An error message stating where error occurred
187 0314 1 GCA_COM_START - A ch$ptr to the start of the command
188 0315 1 FS_NEXT (IRA) - A ch$ptr to the character after the last one
189 0316 1 in the command
190 0317 1
191 0318 1 IMPLICIT OUTPUTS: None
192 0319 1
193 0320 1 ROUTINE VALUE:
194 0321 1 COMPLETION CODES: None
195 0322 1
196 0323 1 SIDE EFFECTS: None
197 0324 1 --
198 0325 1
199 0326 2 BEGIN
200 0327 2
201 0328 2 IF .position
202 0329 2 THEN
203 0330 2 endcmt (); ! Position to end of command.
204 0331 2
205 0332 2 erm (.error, .gca_com_start, CH$DIFF (.fs_next (ira), .gca_com_start));
206 0333 2 erm (rnfloc, 0, 0);
207 0334 1 END; ! End of ERMA
```

.TITLE ERROR Field (error) message requests from other  
modul  
.IDENT \V04-000\

ERROR  
V04-000

Field (error) message requests from other modul 16-Sep-1984 00:25:59  
ERMA -- add DIRECTIVE and location to error mes 14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742 Page 6  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1 (4)

```
.PSECT $OWNS,NOEXE,2

00000 WORK_BUFFER:
01F4 001F4 WORK_DESC:
01 0E 001F6 .WORD 500
00000000 001F8 .BYTE 14, 1
001FC W_BUFFER1:
0082 00280 W_DESC1: .BLKB 132
01 0E 00282 .WORD 130
00000000 00284 .BYTE 14, 1
00284 .ADDRESS W_BUFFER1

.EXTRN GCA, IRA, IRAC, MSGTXT
.EXTRN PAGEN, PHAN, RNOIOB
.EXTRN BRNOOB, SEMCOD, TIOIOB
.EXTRN TTEIOB, RNFLOC, ENDCMT
.EXTRN PACBAS, PACPAG, PACSTR
.EXTRN SYS$GETMSG, EMSG

.PSECT $CODE$,NOWRT,2

000C 00000 .ENTRY ERMA, Save R2,R3
53 00000000V EF 9E 00002 MOVAB ERM, R3
52 00000000G EF 9E 00009 MOVAB GCA, R2
07 08 AC E9 00010 BLBC POSITION, 1$
00000000G EF 00 FB 00014 CALLS #0, ENDCMT
7E 00000000G EF 62 C3 0001B 1$: SUBL3 GCA, IRA+4, -(SP)
04 AC DD 00023 PUSHL GCA
63 03 DD 00025 PUSHL ERROR
03 FB 00028 CALLS #3, ERM
00000000G 7E 7C 0002B CLRQ -(SP)
63 03 DD 0002D PUSHL #RNFLOC
03 FB 00033 CALLS #3, ERM
04 00036 RET
```

; Routine Size: 55 bytes, Routine Base: \$CODE\$ + 0000

; 208 0335 1



```
210 0336 1 %SBTTL 'ERMB -- add input text & location to error message'
211 0337 1 GLOBAL ROUTINE ermb (error,position) : NOVALUE =
212 0338 1
213 0339 1 ++
214 0340 1 FUNCTIONAL DESCRIPTION:
215 0341 1
216 0342 1 This routine contains a sequence of code that is commonly used
217 0343 1 for handling errors.
218 0344 1
219 0345 1 Processing is like ERMA except that the entire input line is
220 0346 1 displayed rather than just a command.
221 0347 1
222 0348 1 If the POSITION parameter is TRUE, it calls ENDCMT to
223 0349 1 position to the end of the erroneous command. It always passes
224 0350 1 to ERM the specified error message (ERROR), and the start and
225 0351 1 length of the string to be inserted into the error message.
226 0352 1 Note that this call on ERM does not depend on the value of POSITION.
227 0353 1
228 0354 1 Finally, it calls ERM to output the error message indicating
229 0355 1 where in the input and output files the error occurred.
230 0356 1
231 0357 1 FORMAL PARAMETERS:
232 0358 1
233 0359 1 ERROR - Address of desired error diagnostic. Passed to ERM.
234 0360 1 POSITION - If TRUE, ERMA positions to the end of the
235 0361 1 command before issuing the second call on ERM.
236 0362 1
237 0363 1 IMPLICIT INPUTS:
238 0364 1
239 0365 1 RNFLOC - An error message stating where error occurred
240 0366 1 FS_START (IRA) - A ch$ptr to the start of the input line.
241 0367 1 FS_MAXSIZE (IRA) - The total size of the input line.
242 0368 1
243 0369 1 IMPLICIT OUTPUTS: None
244 0370 1
245 0371 1 ROUTINE VALUE:
246 0372 1 COMPLETION CODES: None
247 0373 1
248 0374 1 SIDE EFFECTS: None
249 0375 1 --
250 0376 1
251 0377 2 BEGIN
252 0378 2
253 0379 2 IF .position
254 0380 2 THEN
255 0381 2 endcmt (); ! Position to end of command.
256 0382 2
257 0383 2 erm (.error, .fs_start (ira), .fs_maxsize (ira));
258 0384 2 erm (rnfloc, 0, 0);
259 0385 1 END; ! End of ERMB
```

0004 00000  
52 00000000V EF 9E 00002.ENTRY ERMB, Save R2  
MOVAB ERM, R2: 0337  
:

ERROR  
V04-000

Field (error) message requests from other modul 16-Sep-1984 00:25:59  
ERMB -- add input text & location to error mess 14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1

Page 8  
(5)

00000000G 07 08 AC E9 00009  
EF 00 FB 0000D  
00000000G EF DD 00014 1\$:  
00000000G EF DD 0001A  
04 AC DD 00020  
62 03 FB 00023  
7E 7C 00026  
00000000G 8F DD 00028  
62 03 FB 0002E  
04 00031

BLBC POSITION, 1\$  
CALLS #0, ENDCMT  
PUSHL IRA+8  
PUSHL IRA  
PUSHL ERROR  
CALLS #3, ERM  
CLRQ -(SP)  
PUSHL #RNFLOC  
CALLS #3, ERM  
RET

: 0379  
: 0381  
: 0383  
: 0384  
: 0385

; Routine Size: 50 bytes, Routine Base: \$CODE\$ + 0037

; 260 0386 1

ERROR  
V04-000

Field (error) message requests from other modul  
ERME -- report message w/secondary message

L 2  
16-Sep-1984 00:25:59  
14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1  
Page 9  
(6)

```
262 0387 1 %SBTTL 'ERME -- report message w/secondary message'
263 0388 1 GLOBAL ROUTINE erme (error, arg, len, msgcode) : NOVALUE =
264 0389 1
265 0390 1 ++
266 0391 1 FUNCTIONAL DESCRIPTION:
267 0392 1
268 0393 1 This routine will report the message as passed and then print a second
269 0394 1 informational line based on the passed message code (msgcode).
270 0395 1
271 0396 1 FORMAL PARAMETERS:
272 0397 1
273 0398 1 ERROR - Address of desired error diagnostic. Passed to ERM.
274 0399 1 ARG - A CH$PTR to the string to be output.
275 0400 1 LEN - The length of the string.
276 0401 1 MSGCODE - Error message code for the secondary error message.
277 0402 1
278 0403 1 IMPLICIT INPUTS: None
279 0404 1
280 0405 1 IMPLICIT OUTPUTS: None
281 0406 1
282 0407 1 ROUTINE VALUE:
283 0408 1 COMPLETION CODES: None
284 0409 1
285 0410 1 SIDE EFFECTS: None
286 0411 1 --
287 0412 1
288 0413 2 BEGIN
289 0414 2 erm (.error, .arg, .len);
290 0415 2 report_secondary_message (.msgcode);
291 0416 1 END; ! End of ERMS
```

```
0000 0000
7E 08 AC 7D 00002
04 AC DD 00006
00000000V EF 03 FB 00009
10 AC DD 00010
00000000V EF 01 FB 00013
04 0001A
```

```
.ENTRY ERME, Save nothing
MOVQ ARG, -(SP)
PUSHL ERROR
CALLS #3, ERM
PUSHL MSGCODE
CALLS #1, REPORT_SECONDARY_MESSAGE
RET
```

```
0388
0414
0415
0416
```

; Routine Size: 27 bytes, Routine Base: \$CODE\$ + 0069

; 292 0417 1



ERROR  
V04-000

Field (error) message requests from other modul 16-Sep-1984 00:25:59  
ERML -- add location to error message without a 14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742 Page 10  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1 (7)

```

294 0418 1 XSBTTL 'ERML -- add location to error message without additional data'
295 0419 1 GLOBAL ROUTINE erml (error) : NOVALUE =
296 0420 1
297 0421 1 ++
298 0422 1 FUNCTIONAL DESCRIPTION:
299 0423 1
300 0424 1 This routine contains a sequence of code that is commonly used
301 0425 1 for handling errors.
302 0426 1
303 0427 1 ERML is just a shortcut for the following calls:
304 0428 1 ERM (.ERROR, 0, 0);
305 0429 1 ERM (RNFLOC, 0, 0);
306 0430 1 In other words, it saves code.
307 0431 1
308 0432 1
309 0433 1 FORMAL PARAMETERS:
310 0434 1
311 0435 1 ERROR - Address of desired error diagnostic. Passed to ERM.
312 0436 1
313 0437 1 IMPLICIT INPUTS:
314 0438 1
315 0439 1 RNFLOC - An error message stating where error occurred
316 0440 1
317 0441 1 IMPLICIT OUTPUTS: None
318 0442 1
319 0443 1 ROUTINE VALUE:
320 0444 1 COMPLETION CODES: None
321 0445 1
322 0446 1 SIDE EFFECTS: None
323 0447 1 --
324 0448 1
325 0449 2 BEGIN
326 0450 2 erm (.error, 0, 0);
327 0451 2 erm (rnfloc, 0, 0);
328 0452 1 END; ! End of ERML
```

```

52 00000000V EF 9E 00002
7E 7C 00009
04 AC DD 0000B
62 03 FB 0000E
7E 7C 00011
00000000G 8F DD 00013
62 03 FB 00019
04 0001C
```

```

.ENTRY ERML, Save R2
MOVAB ERM, R2
CLRQ -(SP)
PUSHL ERROR
CALLS #3, ERM
CLRQ -(SP)
PUSHL #RNFLOC
CALLS #3, ERM
RET
```

```

: 0419
: 0450
: 0451
: 0452
```

; Routine Size: 29 bytes, Routine Base: \$CODE\$ + 0084

; 329 0453 1

ERROR  
V04-000

Field (error) message requests from other modul N 2  
ERMN -- add location to error message w/numeric 16-Sep-1984 00:25:59  
14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742 Page 11  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1 (8)

ERR0  
V04-

```
0454 1 ZSBTTL 'ERMN -- add location to error message w/numeric data'
0455 1 GLOBAL ROUTINE erm (error, arg) : NOVALUE =
0456 1
0457 1 ++
0458 1 FUNCTIONAL DESCRIPTION:
0459 1
0460 1     This routine contains a sequence of code that is commonly used
0461 1     for handling errors.
0462 1
0463 1     The code is straightforward. It is mainly a code-saver.
0464 1
0465 1
0466 1 FORMAL PARAMETERS:
0467 1
0468 1     ERROR - Address of desired error diagnostic. Passed to ERM.
0469 1     ARG -   A number to be output.
0470 1
0471 1 IMPLICIT INPUTS:
0472 1
0473 1     RNFLOC - An error message stating where error occurred
0474 1
0475 1 IMPLICIT OUTPUTS:      None
0476 1
0477 1 ROUTINE VALUE:
0478 1 COMPLETION CODES:      None
0479 1
0480 1 SIDE EFFECTS:          None
0481 1 --
0482 1
0483 2 BEGIN
0484 2 erm (.error, .arg, 0);
0485 2 erm (rnfloc, 0, 0);
0486 1 END;          ! End of ERMN
```

```
52 00000000V EF 9E 00002
7E 04 AC 7D 00009
62 03 FB 0000F
00000000G 7E 7C 00012
62 8F DD 00014
03 FB 0001A
04 0001D
```

```
.ENTRY ERMN, Save R2
MOVAB ERM, R2
CLRL -(SP)
MOVQ ERROR, -(SP)
CALLS #3, ERM
CLRQ -(SP)
PUSHL #RNFLOC
CALLS #3, ERM
RET
```

```
: 0455
: 0484
: 0485
: 0486
```

; Routine Size: 30 bytes. Routine Base: \$CODE\$ + 00A1

; 364 0487 1

ERROR  
V04-000

Field (error) message requests from other modul 16-Sep-1984 00:25:59  
ERMS -- add location to error message w/string 14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742 Page 12  
DISK\$VM\$MASTER:[RUNOFF.SRC]ERROR.BLI;1 (9)

```

366 0488 1 %SBTTL 'ERMS -- add location to error message w/string data'
367 0489 1 GLOBAL ROUTINE erms (error, arg, len) : NOVALUE =
368 0490 1
369 0491 1 ++
370 0492 1 FUNCTIONAL DESCRIPTION:
371 0493 1
372 0494 1 This routine contains a sequence of code that is commonly used
373 0495 1 for handling errors.
374 0496 1
375 0497 1 The code is straightforward. It is mainly a code-saver.
376 0498 1
377 0499 1
378 0500 1 FORMAL PARAMETERS:
379 0501 1
380 0502 1 ERROR - Address of desired error diagnostic. Passed to ERM.
381 0503 1 ARG - A CH$PTR to the string to be output.
382 0504 1 LEN - The length of the string.
383 0505 1
384 0506 1 IMPLICIT INPUTS:
385 0507 1
386 0508 1 RNFLOC - An error message stating where error occurred
387 0509 1
388 0510 1 IMPLICIT OUTPUTS: None
389 0511 1
390 0512 1 ROUTINE VALUE:
391 0513 1 COMPLETION CODES: None
392 0514 1
393 0515 1 SIDE EFFECTS: None
394 0516 1 --
395 0517 1
396 0518 2 BEGIN
397 0519 2 erm (.error, .arg, .len);
398 0520 2 erm (rnfloc, 0, 0);
399 0521 1 END; ! End of ERMS
```

```

52 00000000V EF 9E 00002
7E 08 AC 7D 00009
04 AC DD 0000D
62 03 FB 00010
7E 7C 00013
00000000G 8F DD 00015
62 03 FB 0001B
04 0001E
```

```

.ENTRY ERMS, Save R2
MOVAB ERM, R2
MOVQ ARG, -(SP)
PUSHL ERROR
CALLS #3, ERM
CLRQ -(SP)
PUSHL #RNFLOC
CALLS #3, ERM
RET
```

```

: 0489
: 0519
: 0520
: 0521
```

; Routine Size: 31 bytes, Routine Base: \$CODE\$ + 00BF

; 400 0522 1



```
402 0523 1 XSBTTL 'ERM -- funnel message to appropriate error output mechanism'
403 0524 1 GLOBAL ROUTINE erm (error, arg, len) : NOVALUE =
404 0525 1
405 0526 1 ++
406 0527 1 FUNCTIONAL DESCRIPTION:
407 0528 1
408 0529 1 This routine generates and prints the requested error diagnostic.
409 0530 1 The first routine argument points to an error string consisting of
410 0531 1 a string intermixed with special formatting flags which cause special
411 0532 1 actions to happen to the string when a flag is encountered.
412 0533 1 The routine SCNMSG is called to expand the error message
413 0534 1 before it is output.
414 0535 1
415 0536 1 FORMAL PARAMETERS:
416 0537 1
417 0538 1 ERROR - Pointer to the desired error diagnostic.
418 0539 1 ARG - Contains a value if it is to be used with %N;
419 0540 1 contains a pointer to a string if it is to be used with %S.
420 0541 1 LEN - Unused with %N, Contains the string length for %S.
421 0542 1
422 0543 1 All parameters are 'pass through' parameters for SCNMSG.
423 0544 1
424 0545 1 IMPLICIT INPUTS: None
425 0546 1
426 0547 1 IMPLICIT OUTPUTS:
427 0548 1
428 0549 1 GCA_FEHLER - Set to TRUE to indicate an error occurred
429 0550 1 GCA_ERRCNT - Count of lines of error messages
430 0551 1
431 0552 1 ROUTINE VALUE:
432 0553 1 COMPLETION CODES: None
433 0554 1
434 0555 1 SIDE EFFECTS: None
435 0556 1 --
436 0557 1
437 0558 1 BEGIN
438 0559 2 EXTERNAL LITERAL !Error messages previously undeclared
439 0560 2 rnfba, I - See command on input line %C of page %I of file '%S'
440 0561 2 rnfdr, I - DSR(PLUS) Version %V: %N diagnostic message%X reported
441 0562 2 rnfmr, W - Another %N crossed margin or bad right indent attempt%X detected and accumulated. Now
442 0563 2 rnfnd, I - DSR(PLUS) Version %V: No errors detected
443 0564 2 rnfnc, W - Another %N negative indent%X detected and accumulated. Now being reported
444 0565 2 rnfpc1, I - Illegal message: rnfpc1
445 0566 2 rnfpc2, I - %N page%X written to '%O'
446 0567 2 rnfstr, I - '%S'
447 0568 2 U U U U %IF DSRPLUS %THEN
448 0569 2 rnfpc4, ! %N cross-reference record%X written to %B
449 0570 2 rnfpc5, ! %N cross-reference record%X read from %A
450 0571 2 rnfcr, ! One or more forward cross-references could not be resolved
451 0572 2 rnfcr1, ! Run DSRPLUS again if you need them correct
452 0573 2 rnfcr2, ! Or run DSRPLUS/DEBUG to see which one(s) changed
453 0574 2 U U U U %FI
454 0575 2 rnfpc2, ! %N indexing record%X written to %B
455 0576 2 rnfpc3, ! %N table-of-contents record%X written to %B
456 0577 2
457 0578 2 LITERAL e_number_mask = %X'7FF8';
458 0579 2
```

```
459      0580      LOCAL
460      0581          line_cnt,          !Size of output line
461      0582          e_number,          !error number
462      0583          status;              !Status code returned by XPORT
463      0584
464      0585      %IF %BLISS(BLISS32) %THEN
465      0586          ! Initialize message string descriptor:
466      0587          work_desc [str$b_dtype] = str$k_dtype_t;      ! ASCII text (8-bit)
467      0588          work_desc [str$b_class] = str$k_class_f;      ! Scalar, String Descriptor
468      0589          work_desc [str$a_pointer] = work_buffer [0]; ! First byte of char. buffer
469      0590      %FI
470      0591
471      0592          e_number = (.error AND e_number_mask)/8;
472      0593
473      0594          !Count every real error message
474      0595          IF (
475      0596              AND .error NEQ rnfbak
476      0597              AND .error NEQ rnfird
477      0598              AND .error NEQ rnfloc
478      0599              AND .error NEQ rnfmrk      ! The parameter passed will be added in later.
479      0600              AND .error NEQ rnfnc      ! The parameter passed will be added in later.
480      0601              AND .error NEQ rnfncd
481      0602              AND .error NEQ rnfpc1
482      0603              AND .error NEQ rnfpc2
483      0604              AND .error NEQ rnfpc3
484      0605          %IF DSRPLUS %THEN
485      0606              AND .error NEQ rnfpc4
486      0607              AND .error NEQ rnfpc5
487      0608              AND .error NEQ rnfcr1
488      0609              AND .error NEQ rnfcr2
489      0610          %FI
490      0611          AND .error NEQ rnfstr )
491      0612          THEN
492      0613              BEGIN
493      0614                  gca_fehler = true;
494      0615                  gca_errcnt = .gca_errcnt + 1;
495      0616              END;
496      0617
497      0618          IF (
498      0619              OR .error EQL rnfmrk      ! These two messages display an accumulated
499      0620              OR .error EQL rnfnc      ! count of errors previously unreported.
500      0621          THEN
501      0622              gca_errcnt = .gca_errcnt + .arg;
502      0623
503      0624          line_cnt = schmsg (CH$PTR (work_buffer), .msgtxt [.e_number], .arg, .len);
504      0625
505      0626          %IF %BLISS(BLISS32) %THEN
506      0627              work_desc [str$h_length] = .line_cnt;      ! Put message length into descriptor
507      0628          %FI
508      0629
509      0630          !The line is now packed correctly, so output it to the requested places.
510      0631          CASE .gca_err_dir FROM report_err_none TO report_err_both OF
511      0632              SET
512      0633                  [report_err_none]:      0;      !Don't bother to output the message
513      0634                  [report_err_file]:
514      0635                      BEGIN
515      0636                          !Report error in output file
```

```

516 0637
517 U 0638 %IF NOT %BLISS(BLISS32) %THEN
518 U 0639     status = $XPO_PUT (IOB = .rnoiob
519 U 0640                          ,string = (.line_cnt, CH$PTR (work_buffer)) );
520 U 0641
521 U 0642     ! Add carriage control information to the end of error message
522 U 0643     status = $XPO_PUT (IOB = .rnoiob
523 U 0644                          ,STRING = (2, CH$PTR (UPLIT (%STRING (%CHAR (
524 U 0645                          %0'15' ,%0'12'))))) );
525 U 0646 %ELSE
526 0647     status = emsg (.error, work_desc, true);
527 0648 %FI
528 0649
529 0650     END;
530 0651
531 0652     [report_err_std]:
532 0553
533 0654     BEGIN                                ! Report error on standard error log
534 0655
535 U 0656 %IF NOT %BLISS(BLISS32) %THEN
536 U 0657     status = $XPO_PUT( IOB = tteiob
537 U 0658                          ,STRING = (.line_cnt, CH$PTR (work_buffer)) );
538 0659 %ELSE
539 0660     status = emsg (.error, work_desc, false);
540 0661 %FI
541 0662
542 0663     END;
543 0664
544 0665     [report_err_both]:
545 0666     BEGIN                                ! Report error both places -- output file first
546 0667
547 U 0668 %IF NOT %BLISS(BLISS32) %THEN
548 U 0669     status = $XPO_PUT (IOB = .rnoiob
549 U 0670                          ,STRING = (.line_cnt, CH$PTR (work_buffer)));
550 U 0671
551 U 0672     !Add carriage control information to the end of error message
552 U 0673     status = $XPO_PUT (IOB = .rnoiob
553 U 0674                          ,STRING = (2, CH$PTR (UPLIT (%STRING (%CHAR (
554 U 0675                          %0'15' , %0'12'))))) );
555 U 0676     !Report error on standard error log
556 U 0677     status = $XPO_PUT (IOB = tteiob
557 U 0678                          ,STRING = (.line_cnt, CH$PTR (work_buffer)));
558 0679 %ELSE
559 0680     status = emsg (.error, work_desc, true);
560 0681 %FI
561 0682
562 0683     END;
563 0684
564 0685     TES;
565 0686
566 0687     END;                                !End of ERM

```

```
.EXTRN  RNFBK,  RNFERD,  RNFMRC
.EXTRN  RNFNED,  RNFNIC,  RNFP1
.EXTRN  RNFPCT,  RNFSTR,  RNFP2
```



				007C	00000	.ENTRY	ERM, Save R2,R3,R4,R5,R6		0524
		56	00000000G	8F	D0	00002	MOVL	#RNFNIC, R6	
		55	00000000G	8F	D0	00009	MOVL	#RNFMRC, R5	
		54	00000000G	EF	9E	00010	MOVAB	GCA+196, R4	
		53	00000000'	EF	9E	00017	MOVAB	WORK_DESC, R3	
	02	A3	010E	8F	B0	0001E	MOVW	#270, WORK_DESC+2	0587
	04	A3	FE0C	C3	9E	00024	MOVAB	WORK_BUFFER, WORK_DESC+4	0589
		52	04	AC	D0	0002A	MOVL	ERROR, R2	0592
50		52	FFFF8007	8F	CB	0002E	BICL3	#-32761, R2, R0	
		50		08	C6	00036	DIVL2	#8, E_NUMBER	
	00000000G	8F		52	D1	00039	CMPL	R2, #RNFBK	0595
				58	13	00040	BEQL	1\$	
	00000000G	8F		52	D1	00042	CMPL	R2, #RNFERD	0596
				4F	13	00049	BEQL	1\$	
	00000000G	8F		52	D1	00048	CMPL	R2, #RNFLC	0597
				46	13	00052	BEQL	1\$	
		55		52	D1	00054	CMPL	R2, R5	0598
				41	13	00057	BEQL	1\$	
		56		52	D1	00059	CMPL	R2, R6	0599
				3C	13	0005C	BEQL	1\$	
	00000000G	8F		52	D1	0005E	CMPL	R2, #RNFNED	0600
				33	13	00065	BEQL	1\$	
	00000000G	8F		52	D1	00067	CMPL	R2, #RNFPCT	0601
				2A	13	0006E	BEQL	1\$	
	00000000G	8F		52	D1	00070	CMPL	R2, #RNFP1	0602
				21	13	00077	BEQL	1\$	
	00000000G	8F		52	D1	00079	CMPL	R2, #RNFP2	0603
				18	13	00080	BEQL	1\$	
	00000000G	8F		52	D1	00082	CMPL	R2, #RNFP3	0604
				0F	13	00089	BEQL	1\$	
	00000000G	8F		52	D1	0008B	CMPL	R2, #RNFSR	0612
				06	13	00092	BEQL	1\$	
	FC	A4		01	D0	00094	MOVL	#1, GCA+192	0615
				64	D6	00098	INCL	GCA+196	0616
		55		52	D1	0009A	CMPL	R2, R5	0619
				05	13	0009D	BEQL	2\$	
		56		52	D1	0009F	CMPL	R2, R6	0620
				04	12	000A2	BNEQ	3\$	
	64	08		AC	C0	000A4	ADDL2	ARG, GCA+196	0622
	7E	08		AC	7D	000A8	MOVQ	ARG, -(SP)	0624
			00000000GEF	40	DD	000AC	PUSHL	MSG[XT[E_NUMBER]]	
			FE0C	C3	9F	000B3	PUSHAB	WORK_BUFFER	
	00000000V	EF		04	FB	000B7	CALLS	#4, SCNMSG	
		63		50	B0	000BE	MOVW	LINE_CNT, WORK_DESC	0627
	03	00		A4	CF	000C1	CASEL	GCA+200, #0, #3	0631
000D	0009	000D		0018		000C6	.WORD	8\$-4\$, -	
								6\$-4\$, -	
								5\$-4\$, -	
								6\$-4\$	
				04	000CE	RET			
				7E	D4	000CF	CLRL	-(SP)	0660
				02	11	000D1	BRB	7\$	
				01	DD	000D3	PUSHL	#1	0680
				0C	BB	000D5	PUSHR	#*M<R2,R3>	
	00000000G	EF		03	FB	000D7	CALLS	#3, EMSG	

ERROR  
V04-000

Field (error) message requests from other modul 6 3  
ERM -- funnel message to appropriate error outp 16-Sep-1984 00:25:59  
14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742 Page 17  
DISK\$VMMASTER:[RUNOFF.SRC]ERROR.BLI;1 (10)

04 000DE 8\$: RET

: 0687

; Routine Size: 223 bytes, Routine Base: \$CODE\$ + 00DE

; 567 0688 1

FCI

```
569 0689 1 %SBTTL 'PUTMSG -- funnel message to appropriate output mechanism'
570 0690 1 GLOBAL ROUTINE putmsg (message, arg, len) : NOVALUE =
571 0691 1
572 0692 1 ++
573 0693 1 FUNCTIONAL DESCRIPTION:
574 0694 1
575 0695 1 This routine generates and prints the requested message.
576 0696 1 The first routine argument points to a string consisting of
577 0697 1 a string intermixed with special formatting flags which cause special
578 0698 1 actions to happen to the string when a flag is encountered.
579 0699 1 The routine SCNMSG is called to expand the message
580 0700 1 before it is output.
581 0701 1
582 0702 1 FORMAL PARAMETERS:
583 0703 1
584 0704 1 MESSAGE - Pointer to the desired message.
585 0705 1 ARG - Contains a value, if it is to be used with %N,
586 0706 1 contains a pointer to a string if it is to be used with %S.
587 0707 1 LEN - Unused with %N, Contains the string length for %S.
588 0708 1
589 0709 1 All parameters are 'pass through' parameters for SCNMSG.
590 0710 1
591 0711 1 IMPLICIT INPUTS: None
592 0712 1
593 0713 1 IMPLICIT OUTPUTS: None
594 0714 1
595 0715 1 ROUTINE VALUE:
596 0716 1 COMPLETION CODES: None
597 0717 1
598 0718 1 SIDE EFFECTS: None
599 0719 1 --
600 0720 1
601 0721 2 BEGIN
602 0722 2 LITERAL
603 0723 2 e_number_mask = %X'7FF8';
604 0724 2 LOCAL
605 0725 2 line_cnt, !Size of output line
606 0726 2 e_number, !error number
607 0727 2 status; !Status code returned by XPORT
608 0728 2
609 0729 2 %IF %BLISS(BLISS32) %THEN
610 0730 2 ! Initialize message string descriptor:
611 0731 2
612 0732 2 work_desc [str$b_dtype] = str$k_dtype_t; ! ASCII text (8-bit)
613 0733 2 work_desc [str$b_class] = str$k_class_f; ! Scalar, String Descriptor
614 0734 2 work_desc [str$a_pointer] = work_buffer [0]; ! First byte of char. buffer
615 0735 2 %FI
616 0736 2
617 0737 2 e_number = (.message AND e_number_mask) / 8;
618 0738 2 line_cnt = scnmsg (CH$PTR (work_buffer), .msgtxt [.e_number], .arg, .len);
619 0739 2
620 0740 2 %IF %BLISS(BLISS32) %THEN
621 0741 2 work_desc [str$h_length] = .line_cnt; ! Put message length into descriptor
622 0742 2 %FI
623 0743 2
624 0744 2 !The line is now packed correctly, so output it to the terminal
625 0745 2
```



ERROR  
V04-000

Field (error) message requests from other modul 16-Sep-1984 00:25:59  
PUTMSG -- funnel message to appropriate output 14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742 Page 19  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1 (11)

```

: 626 U 0746 2 %IF NOT %BLISS(BLISS32) %THEN
: 627 U 0747 2 status = $XPD_PUT (IOB = ttojob !Standard message device (usually TTY:)
: 628 U 0748 2 ,STRING = (.line_cnt, CH$PTR (work_buffer)));
: 629 U 0749 2 %ELSE
: 630 0750 2 status = msg (.message, work_desc, false);
: 631 0751 2 %FI
: 632 0752 2
: 633 0753 1 END; !End of PUTMSG
```

```

02 52 00000000' EF 9E 00000
04 A2 010E 8F 80 00009
04 A2 FE0C C2 9E 0000F
50 AC FFFF8007 8F CB 00015
7E 08 AC 7D 00021
00000000GEF40 DD 00025
FE0C C2 9F 0002C
00000000V EF 04 FB 00030
62 50 80 00037
7E D4 0003A
52 DD 0003C
04 AC DD 0003E
00000000G EF 03 FB 00041
04 00048
```

```

.ENTRY PUTMSG, Save R2
MOVAB WORK_DESC, R2
MOVW #270, WORK_DESC+2
MOVAB WORK_BUFFER, WORK_DESC+4
BICL3 #-32761, MESSAGE, -R0
DIVL2 #8, E_NUMBER
MOVQ ARG, -(SP)
PUSHL MSGT[E_NUMBER]
PUSHAB WORK_BUFFER
CALLS #4, SCNMSG
MOVW LINE_CNT, WORK_DESC
CLRL -(SP)
PUSHL R2
PUSHL MESSAGE
CALLS #3, EMSG
RET
```

```

: 0690
: 0732
: 0734
: 0737
: 0738
: 0741
: 0750
: 0753
```

; Routine Size: 73 bytes, Routine Base: \$CODE\$ + 01BD

; 634 0754 1

```
0755 1 %SBTTL 'SCNMSG -- fill in missing arguments in message string'
0756 1 ROUTINE scnmsg (line, message, arg, len) =
0757 1
0758 1 ++
0759 1 FUNCTIONAL DESCRIPTION:
0760 1
0761 1 This routine expands the specified message. The first routine
0762 1 argument is a ch$ptr to a work area where the expanded message is
0763 1 written to. The second routine argument points to a string consisting
0764 1 of a string intermixed with special formatting flags which cause special
0765 1 actions to happen to the string when a flag is encountered. The legal
0766 1 flags in the string are:
0767 1
0768 1 XS - Using the second and third call arguments place the
0769 1 specified string at this point.
0770 1 XN - Using the second argument only, place the specified
0771 1 decimal number at this point.
0772 1 XP - Insert the Current Output Page.
0773 1 XL - Insert the Current Output Line.
0774 1 XI - Insert the input page number.
0775 1 XC - Insert the input sequence number.
0776 1 XF - Insert the name of the current input file.
0777 1 XO - Insert the name of the output file.
0778 1 XA - Insert the name of the old (input) .BRN file.
0779 1 XB - Insert the name of the new (output) .BRN file.
0780 1 XV - Insert the program version number
0781 1 XX - Insert 's' if XN is other than 1.
0782 1
0783 1 More than one of the above arguments can occur in a message,
0784 1 except that XS and XN are mutually exclusive and cannot occur
0785 1 in the same message.
0786 1
0787 1 An illegal flag will be interpreted as plaintext.
0788 1
0789 1 FORMAL PARAMETERS:
0790 1
0791 1 LINE - Ch$ptr to where the message is to be built up
0792 1 MESSAGE - Pointer to the desired message (unexpanded).
0793 1 ARG - Contains a value, if it is to be used with XN,
0794 1 contains a pointer to a string if it is to be used with XS.
0795 1 LEN - Unused with XN, Contains the string length for XS.
0796 1
0797 1 IMPLICIT INPUTS:
0798 1
0799 1 PAGEN - Output Page number
0800 1 PHAN_LINES TP - Output Line number
0801 1 IRAC_IPAGEN - Input page number
0802 1 IRAC_ISEQN - input line count
0803 1 IRAC_FSPECP - Pointer to Input file name string
0804 1 IRAC_FSPECC - Length of Input file name string
0805 1
0806 1 IMPLICIT OUTPUTS: None
0807 1
0808 1 ROUTINE VALUE:
0809 1
0810 1 Returns the number of characters in the expanded message.
0811 1
```

```
693 0812 1 ! SIDE EFFECTS:      None
694 0813 1 !--
695 0814 1
696 0815 1 BEGIN
697 0816 1 LOCAL
698 0817 1     line_cnt,           !Size of output line
699 0818 1     line_ptr,         !String pointer to output line
700 0819 1     strg_cnt,          !Size of input line
701 0820 1     strg_ptr,         !String pointer to input line
702 0821 1
703 0822 1 !Set up for processing
704 0823 1 line_ptr=.line;
705 0824 1 line_cnt=0;
706 0825 1 strg_ptr=.message;
707 0826 1
708 0827 1 !Get input line size (length of orginal error message string)
709 0828 1 strg_cnt = CH$RCHAR_A (strg_ptr);
710 0829 1
711 0830 1 !Now process the entire input string
712 0831 1 INCR i FROM 1 TO .strg_cnt DO
713 0832 1 BEGIN
714 0833 1 LOCAL
715 0834 1     character;
716 0835 1
717 0836 1     character = CH$RCHAR_A (strg_ptr);
718 0837 1
719 0838 1 IF .character NEQ %C'X'
720 0839 1 THEN
721 0840 1 BEGIN           ! Normal text characters are packed directly
722 0841 1     CH$WCHAR_A (.character, line_ptr);
723 0842 1     line_cnt = .line_cnt + 1
724 0843 1 END
725 0844 1 ELSE
726 0845 1 BEGIN           ! Special case flag handling goes on here
727 0846 1     character = CH$RCHAR_A (strg_ptr);
728 0847 1     i=.i+1;
729 0848 1
730 0849 1 !Process all of the alternatives
731 0850 1 SELECTONE .character OF
732 0851 1     SET
733 0852 1
734 0853 1 %IF DSRPLUS %THEN
735 0854 1     [%C'A']:      ! Input .BRN File Specification
736 0855 1     BEGIN
737 0856 1     BIND
738 0857 1         file_spec_descr = brniob [iob$t_resultant] : $STR_DESCRIPTOR ();
739 0858 1
740 0859 1         line_cnt = .line_cnt +
741 0860 1             pacstr (.file_spec_descr [str$a_pointer]
742 0861 1                 ,.file_spec_descr [str$h_length]
743 0862 1                 ,line_ptr);
744 0863 1     END;
745 0864 1 %FI
746 0865 1 [%C'B']:      ! Output .BRN File Specification
747 0866 1     BEGIN
748 0867 1     BIND
749 0868 1         file_spec_descr = brnoob [iob$t_resultant] : $STR_DESCRIPTOR ();
```



```
750 0869 S
751 0870 S
752 0871 S
753 0872 S
754 0873 S
755 0874 S
756 0875 S
757 0876 S
758 0877 S
759 0878 S
760 0879 S
761 0880 S
762 0881 S
763 0882 S
764 0883 S
765 0884 S
766 0885 S
767 0886 S
768 0887 S
769 0888 S
770 0889 S
771 0890 S
772 0891 S
773 0892 S
774 0893 S
775 0894 S
776 0895 S
777 0896 S
778 0897 S
779 0898 S
780 0899 S
781 0900 S
782 0901 S
783 0902 S
784 0903 S
785 0904 S
786 0905 S
787 0906 S
788 0907 S
789 0908 S
790 0909 S
791 0910 S
792 0911 S
793 0912 S
794 0913 S
795 0914 S
796 0915 S
797 0916 S
798 0917 S
799 0918 S
800 0919 S
801 0920 S
802 0921 S
803 0922 S
804 0923 S
805 0924 S
806 0925 S

      line_cnt = .line_cnt +
      pacstr (.file_spec_descr [str$a_pointer]
      ,.file_spec_descr [str$h_length]
      ,line_ptr);

    END;

[XC'C']:      ! Input Sequence Number
      line_cnt = .line_cnt + pacbas (.irac_iseqn, line_ptr, 10);

[XC'F']:      ! Current Input File Name
      line_cnt = .line_cnt + pacstr (.irac_fspecc
      ,.irac_fspecc
      ,line_ptr);

[XC'O']:      ! Output File Specification
    BEGIN
    BIND
      file_spec_descr = rnojob [lob$t_resultant] : $STR_DESCRIPTOR ();

      line_cnt = .line_cnt +
      pacstr (.file_spec_descr [str$a_pointer]
      ,.file_spec_descr [str$h_length]
      ,line_ptr);

    END;

[XC'I']:      ! Input Page Number
      line_cnt = .line_cnt + pacbas (.irac_ipagen, line_ptr, 10);

[XC'L']:      ! Output Line Number
      line_cnt = .line_cnt + pacbas (.phan_lines_tp, line_ptr, 10);

[XC'N']:      ! Numeric passed argument
      line_cnt = .line_cnt + pacbas (.arg, line_ptr, 10);

[XC'P']:      ! Current Output Page Number
      line_cnt = .line_cnt + pacpag (pagen, line_ptr);

[XC'S']:      ! String passed argument
      ! In no case insert more than 100 characters.
      ! (Defensive coding.)

      line_cnt = .line_cnt + pacstr (.arg
      ,MIN (100, .len)
      ,line_ptr);

[XC'V']:      ! Insert version number of program
    BEGIN
    EXTERNAL
      rnovrl,      ! Length of version number string
      rnovrp;      ! CH$PTR to string

      line_cnt = .line_cnt + pacstr (.rnovrp, .rnovrl, line_ptr);
    END;

[XC'X']:      ! Insert an 's' if XN was other than 1.
```

ERROR  
V04-000

Field (error) message requests from other modul 16-Sep-1984 00:25:59  
SCNMSG -- fill in missing arguments in message 14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742 Page 23  
DISK9VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1 (12)

```

807 0926 4      IF .arg NEQ 1
808 0927 4      THEN
809 0928 4      BEGIN
810 0929 4      CH$WCHAR_A (%C's', line_ptr);
811 0930 4      line_cnt=.line_cnt+1
812 0931 4      END;
813 0932 4
814 0933 4      [OTHERWISE]:      ! Unrecognized flag, treat as text
815 0934 4      BEGIN
816 0935 4      CH$WCHAR_A (%C'X', line_ptr);
817 0936 4      CH$WCHAR_A (.character, line_ptr);
818 0937 4      line_cnt = .line_cnt+2
819 0938 4      END;
820 0939 4
821 0940 4      TES
822 0941 4
823 0942 4      END
824 0943 4
825 0944 4      END;
826 0945 4
827 0946 4      RETURN .line_cnt;
828 0947 1      END;      !End of SCNMSG
```

```

                                .EXTRN  RNOVRL, RNOVRP
                                00FC 00000 SCNMSG: .WORD  Save R2,R3,R4,R5,R6,R7
57 00000000G EF 9E 00002 MOVAB  IRAC+8, R7
      04 AC DD 00009 PUSHL  LINE
      52 D4 0000C CLRL   LINE CNT
55      08 AC D0 0000E MOVL   MESSAGE, STRG_PTR
56      85 9A 00012 MOVZBL (STRG_PTR)+, STRG_CNT
      54 D4 00015 CLRL   I
      0135 31 00017 BRW    21$
53      85 9A 0001A 1$: MOVZBL (STRG_PTR)+, CHARACTER
25      53 D1 0001D CMPL   CHARACTER, #37
      07 13 00020 BEQL   2$
00 BE      53 90 00022 MOVB   CHARACTER, @LINE_PTR
      0111 31 00026 BRW    19$
53      85 9A 00029 2$: MOVZBL (STRG_PTR)+, CHARACTER
00000042 8F      54 D6 0002C INCL   I
      53 D1 0002E CMPL   CHARACTER, #66
      11 12 00035 BNEQ   3$
      5E DD 00037 PUSHL  SP
7E 00000000G EF 3C 00039 MOVZWL FILE_SPEC_DESCR, -(SP)
00000000G EF DD 00040 PUSHL  FILE_SPEC_DESCR+4
      3C 11 00046 BRB    6$
00000043 8F      53 D1 00048 3$: CMPL   CHARACTER, #67
      09 12 0004F BNEQ   4$
      0A DD 00051 PUSHL  #10
      04 AE 9F 00053 PUSHAB LINE_PTR
      67 DD 00056 PUSHL  IRAC+8
      66 11 00058 BRB    10$
00000046 8F      53 D1 0005A 4$: CMPL   CHARACTER, #70
      08 12 00061 BNEQ   5$
      5E DD 00063 PUSHL  SP
```

ERROR  
V04-000

Field (error) message requests from other modul  
SCNMSG -- fill in missing arguments in message

N 3  
16-Sep-1984 00:25:59  
14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742

DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1 (12)

Page 24

FCI  
V04-

	7E	08	A7	7D	00065	MOVQ	IRAC+16, -(SP)		
			19	11	00069	BRB	6\$		
0000004F	8F		53	D1	0006B	5\$:	CMPL	CHARACTER, #79	0884
			12	12	00072		BNEQ	7\$	
50 00000000G	EF		1C	C1	00074		ADDL3	#28, RNOIOB, R0	0887
			5E	DD	0007C		PUSHL	SP	0890
	7E		60	3C	0007E		MOVZWL	(R0), -(SP)	0891
		04	A0	DD	00081		PUSHL	4(R0)	0890
			7B	11	00084	6\$:	BRB	14\$	
00000049	8F		53	D1	00086	7\$:	CMPL	CHARACTER, #73	0895
			0A	12	0008D		BNEQ	8\$	
			0A	DD	0008F		PUSHL	#10	0896
		04	AE	9F	00091		PUSHAB	LINE_PTR	
		04	A7	DD	00094		PUSHL	IRACT+12	
			27	11	00097		BRB	10\$	
0000004C	8F		53	D1	00099	8\$:	CMPL	CHARACTER, #76	0898
			0D	12	000A0		BNEQ	9\$	
			0A	DD	000A2		PUSHL	#10	0899
		04	AE	9F	000A4		PUSHAB	LINE_PTR	
		00000000G	EF	DD	000A7		PUSHL	PHANT+12	
			11	11	000AD		BRB	10\$	
0000004E	8F		53	D1	000AF	9\$:	CMPL	CHARACTER, #78	0901
			11	12	000B6		BNEQ	11\$	
			0A	DD	000B8		PUSHL	#10	0902
		04	AE	9F	000BA		PUSHAB	LINE_PTR	
		0C	AC	DD	000BD		PUSHL	ARG	
00000000G	EF		03	FB	000C0	10\$:	CALLS	#3, PACBAS	
			58	11	000C7		BRB	17\$	
00000050	8F		53	D1	000C9	11\$:	CMPL	CHARACTER, #80	0904
			11	12	000D0		BNEQ	12\$	
			5E	DD	000D2		PUSHL	SP	0905
		00000000G	EF	9F	000D4		PUSHAB	PAGEN	
00000000G	EF		02	FB	000DA		CALLS	#2, PACPAG	
			3E	11	000E1		BRB	17\$	
00000053	8F		53	D1	000E3	12\$:	CMPL	CHARACTER, #83	0907
			17	12	000EA		BNEQ	15\$	
			5E	DD	000EC		PUSHL	SP	0912
		10	AC	DD	000EE		PUSHL	LEN	0913
00000064	8F		6E	D1	000F1		CMPL	(SP), #100	
			04	15	000F8		BLEQ	13\$	
	6E	64	8F	9A	000FA		MOVZBL	#100, (SP)	
		0C	AC	DD	000FE	13\$:	PUSHL	ARG	0912
			17	11	00101	14\$:	BRB	16\$	
00000056	8F		53	D1	00103	15\$:	CMPL	CHARACTER, #86	0916
			1A	12	0010A		BNEQ	18\$	
			5E	DD	0010C		PUSHL	SP	0922
		00000000G	EF	DD	0010E		PUSHL	RNOVRL	
		00000000G	EF	DD	00114		PUSHL	RNOVRP	
00000000G	EF		03	FB	0011A	16\$:	CALLS	#3, PACSTR	
	52		50	CO	00121	17\$:	ADDL2	R0, LINE_CNT	
			29	11	00124		BRB	21\$	0850
00000058	8F		53	D1	00126	18\$:	CMPL	CHARACTER, #88	0925
			11	12	0012D		BNEQ	20\$	
	01	0C	AC	D1	0012F		CMPL	ARG, #1	0926
			1A	13	00133		BEQL	21\$	
00	BE	73	8F	90	00135		MOVB	#115, @LINE_PTR	0929
			6E	D6	0013A	19\$:	INCL	LINE_PTR	



ERROR  
V04-000

Field (error) message requests from other modul  
SCNMSG -- fill in missing arguments in message

B 4  
16-Sep-1984 00:25:59  
14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1

Page 25  
(12)

			52	D6	0013C		INCL	LINE_CNT	:	0930
			0F	11	0013E		BRB	21\$	:	0926
	00	BE	25	90	00140	20\$:	MOVB	#37, @LINE_FTR	:	0935
			6E	D6	00144		INCL	LINE_PTR	:	
	00	BE	53	90	00146		MOVB	CHARACTER, @LINE_PTR	:	0936
			6E	D6	0014A		INCL	LINE_PTR	:	
FEC5			02	C0	0014C		ADDL2	#2, LINE_CNT	:	0937
	54		56	F1	0014F	21\$:	ACBL	STRG_CNT; #1, 1, 1\$	:	0838
		52	52	D0	00155		MOVL	LINE_CNT, R0	:	0946
		01	04	00158			RET		:	0947
		50							:	

; Routine Size: 345 bytes, Routine Base: \$CODE\$ + 0206

; 829 0948 1

ERROR  
V04-000

Field (error) message requests from other modul 16-Sep-1984 00:25:59  
REPORT\_SECONDARY\_MESSAGE -- if present, get and 14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742 Page 26  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1 (13)

```
0949 1 XSBTTL 'REPORT_SECONDARY_MESSAGE -- if present, get and output it'
0950 1 ROUTINE report_secondary_message (msgcode) =
0951 1
0952 1 ++
0953 1 FUNCTIONAL DESCRIPTION:
0954 1
0955 1     If the MSGCODE is nonzero, it identifies an error message. Capture
0956 1     that error message and report it using a call to ERM.
0957 1
0958 1     This secondary error message handler was implemented to get the
0959 1     detailed reason for issuing the following error messages:
0960 1
0961 1     RNFCOB, RNFCOF, RNFCOI, RNFCOO, RNFCOR, RNFCOT, or RNFCOX.
0962 1
0963 1 FORMAL PARAMETERS:
0964 1
0965 1     MSGCODE - An error message code. For the time being, we assume it's
0966 1     within the scope of XPORT's error messages.
0967 1
0968 1 IMPLICIT INPUTS:      None
0969 1
0970 1 IMPLICIT OUTPUTS:    None
0971 1
0972 1 ROUTINE VALUE:
0973 1
0974 1     Returns TRUE if the message was there and reported.
0975 1     Returns FALSE if the code was zero or otherwise not reported.
0976 1
0977 1 SIDE EFFECTS:        None
0978 1
0979 1 --
0980 2 BEGIN
0981 2 EXTERNAL LITERAL
0982 2     rnfstr; ! I - 'XS'
0983 2
0984 2 ! Initialize message string descriptor:
0985 2     w_desc1 [str$b_dtype] = str$k_dtype_t;      ! ASCII text (8-bit)
0986 2     w_desc1 [str$b_class] = str$k_class_f;      ! Fixed (Scalar) String Descriptor
0987 2     w_desc1 [str$a_pointer] = w_buffer1[0];     ! First byte of char. buffer
0988 2     w_desc1 [str$h_length] = 130;               ! Size of buffer
0989 2
0990 2 IF .msgcode NEQ 0
0991 2 THEN ! If there is at least something in the code,
0992 2     BEGIN ! attempt to capture the message and report it.
0993 2
0994 2 XIF XBLISS(BLISS32) XTHEN
0995 2     sys$getmsg (.msgcode
0996 2                 .w_desc1 [str$h_length]
0997 2                 .w_desc1
0998 2                 .1
0999 2                 .0 );
1000 2
1001 2 XELSE
1002 2     XPOSXMSG (.msgcode, w_desc1 );      ! Get message from XPORT
1003 2 XFI
1004 2
1005 2     erm (RNFSTR, .w_desc1 [str$a_pointer], .w_desc1 [str$h_length]);
```

ERROR  
V04-000

Field (error) message requests from other modul  
REPORT\_SECONDARY\_MESSAGE -- if present, get and

D 4

16-Sep-1984 00:25:59  
14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[RUNOFF.SRC]ERROR.BLI;1

Page 27  
(13)

```

: 888      1006 3      RETURN TRUE;      ! Return code indicating that a
: 889      1007 3      END
: 890      1008 3      ELSE
: 891      1009 3      RETURN FALSE;      ! Return code indicating that a
: 892      1010 3      ! message has NOT been output.
: 893      1011 1      END;      !End of SCNMSG
```

```

                                0004 00000 REPORT_SECONDARY MESSAGE:
                                .WORD      Save R2
                                52 00000000' EF 9E 00002      MOVAB      W_DESC1, R2
                                62 010E0082 8F D0 00009      MOVL      #T7694850, W_DESC1
                                04 A2      FF7C C2 9E 00010      MOVAB      W_BUFFER1, W_DESC1+4
                                04      04 AC D5 00016      TSTL      MSGCODE
                                7E      01 7D 00019      BEQL      1$
                                52 DD 0001E      MOVQ      #1, -(SP)
                                52 DD 00020      PUSHL     R2
                                04 AC DD 00022      PUSHL     R2
                                00000000G EF 05 FB 00025      PUSHL     MSGCODE
                                7E      62 3C 0002C      CALLS     #5, SYS$GETMSG
                                04 A2 DD 0002F      MOVZWL    W_DESC1, -(SP)
                                00000000G 8F DD 00032      PUSHL     W_DESC1+4
                                FD42 CF 03 FB 00038      PUSHL     #RNFSTR
                                50      01 D0 0003D      CALLS     #3, ERM
                                04 00040      MOVL      #1, R0
                                50 D4 00041 1$:      RET
                                04 00043      CLRL      R0
                                RET
                                : 0950
                                : 0988
                                : 0987
                                : 0990
                                : 0996
                                : 1005
                                : 1009
                                : 1011
```

; Routine Size: 68 bytes, Routine Base: \$CODE\$ + 035F

```

: 894      1012 1
: 895      1013 1 END      !End of module
: 896      1014 0 ELUDOM
```

#### PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	648	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	931	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

#### Library Statistics

----- Symbols ----- Pages Processing



ERROR  
V04-000

Field (error) message requests from other modul  
REPORT\_SECONDARY\_MESSAGE -- if present, get and

E 4  
16-Sep-1984 00:25:59  
14-Sep-1984 13:06:10

VAX-11 Bliss-32 V4.0-742  
DISK\$VMMASTER:[RUNOFF.SRC]ERROR.BLI;1 (13)

Page 28

File	Total	Loaded	Percent	Mapped	Time
\$255\$DUA28:[SYSLIB]XPORT.L32;1	590	89	15	252	00:00.2
\$255\$DUA28:[RUNOFF.SRC]DSRLIB.L32;1	1248	29	2	86	00:00.3

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS\$ :ERROR/OBJ=OBJ\$ :ERROR MSRC\$ :ERROR/UPDATE=(ENH\$ :ERROR)

: Size: 931 code + 648 data bytes  
: Run Time: 00:19.9  
: Elapsed Time: 00:53.5  
: Lines/CPU Min: 3055  
: Lexemes/CPU-Min: 28149  
: Memory Used: 149 pages  
: Compilation Complete



0341

AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY